Exactly solving TSP using the Simplex algorithm

Andrej Ivašković, Thomas Sauerwald

CST Part II
ADVANCED ALGORITHMS

15 May 2019

(original slides by Petar Veličković)
**Travelling Salesman Problem**

- **Brute-force solution:** $O(n!)$
- **Dynamic programming algorithms:** $O(n^2 2^n)$
- **Selling on eBay:** $O(1)$

Still working on your route?

Shut the hell up.
Aside: Held–Karp algorithm

- Use a *dynamic programming* approach. *Main idea*: solve the slightly simpler problem of the shortest *path* visiting all nodes, then route the end to the beginning.

- Assume (wlog) that the path starts from node 1. Given a node $x$ and set of nodes $S$ with $1 \in S$, maintain the solution $dp(x, S)$ as the shortest path length starting from 1, visiting all nodes in $S$, and ending in $x$.

- Base case: $dp(1, \{1\}) = 0$.

- Recurrence relation:

\[
dp(x, S) = \begin{cases} 
\min_{y \in S} \{dp(y, S \setminus \{x\}) + c_{yx}\} & x \in S \land 1 \in S \\
+\infty & \text{otherwise}
\end{cases}
\]
Aside: Held–Karp algorithm

- Finally, $dp(x, V)$ will give the shortest path visiting all nodes, starting in 1 and ending in $x$.
- Now the optimum TSP length is simply:

$$\min_{x \in V} \{dp(x, V) + c_{x1}\}$$

The cycle itself can be extracted by backtracking.
- The set $S$ can be efficiently maintained as an $n$-bit number, with the $i$-th bit indicating whether or not the $i$-th node is in $S$.
- Complexity: $O(n^22^n)$ time, $O(n2^n)$ space.
We will be using *indicator variables* \( x_{ij} \), which should be set to 1 if the edge \( i \leftrightarrow j \) is included in the optimum cycle, and 0 otherwise. To avoid duplication, we impose \( i > j \).

An adequate linear program is as follows:

\[
\begin{align*}
\text{minimise} & \quad \sum_{i=1}^{n} \sum_{j=1}^{i-1} c_{ij} x_{ij} \\
\text{subject to} & \quad \sum_{j<i} x_{ij} + \sum_{j>i} x_{ji} = 2 \\
& \quad \forall i, 1 \leq j < i \leq n \quad x_{ij} \leq 1 \\
& \quad \forall i, 1 \leq j < i \leq n \quad x_{ij} \geq 0
\end{align*}
\]

This is *intentionally* an incompletely specified problem:

- We allow for *subcycles* in the returned path.
- We allow for “partially used edges” \((0 < x_{ij} < 1)\) – this LP approximates an integer program.
If the Simplex algorithm finds a correct cycle (with no subcycles or partially used edges) on the underspecified LP instance, then we have successfully solved the problem!

Otherwise, we need to resort to further specifying the problem by adding additional constraints (manually or automatically).
Further constraints: subcycles

- If the returned solution contains a subcycle, we may eliminate it by adding an explicit constraint against it, and then attempt solving the LP again.

- For a subcycle containing nodes from a set $S \subset V$, we may demand at least two edges between $S$ and $V \setminus S$:

$$\sum_{\substack{i \in S \\
j \in V \setminus S}} x_{\max(i,j), \min(i,j)} \geq 2$$

- We will not add all of these constraints – why?

- We often don’t need to add all the constraints in order to reach a valid solution.
Further constraints: partially used edges

- If the returned solution contains a partially used edge, we may attempt a *branch&bound* strategy on it.

- For a partially used edge $a \leftrightarrow b$, we initially add a constraint $x_{ab} = 1$, and continue solving the LP.

- Once a valid solution has been found, we remove all the constraints added since then, add a new constraint $x_{ab} = 0$, and solve the LP again.

- We may stop searching a branch if we reach a worse objective value than the best valid solution found so far.

- The optimum solution is the better out of the two obtained solutions! If we choose the edges wisely, we may often obtain a valid solution in a complexity much better than exponential.
SOLUTION OF A LARGE-SCALE TRAVELING-SALESMAN PROBLEM*

G. DANTZIG, R. FULKERSON, AND S. JOHNSON

The Rand Corporation, Santa Monica, California

(Received August 9, 1954)

It is shown that a certain tour of 49 cities, one in each of the 48 states and Washington, D. C., has the shortest road distance.

The TRAVELING-SALESMAN PROBLEM might be described as follows: Find the shortest route (tour) for a salesman starting from a given city, visiting each of a specified group of cities, and then returning to the original point of departure. More generally, given an $n$ by $n$ symmetric matrix $D = (d_{ij})$, where $d_{ij}$ represents the ‘distance’ from $I$ to $J$, arrange the points in a cyclic order in such a way that the sum of the $d_{ij}$ between consecutive points is minimal. Since there are only a finite number of possibilities (at most $\frac{1}{2} (n-1)!$) to consider, the problem is to devise a method of picking out the optimal arrangement which is reasonably efficient for fairly large values of $n$. Although algorithms have been devised for problems of similar nature, e.g., the optimal assignment problem, little is known about the traveling-salesman problem. We do not claim that this note alters the situation very much; what we shall do is outline a way of approaching the problem that sometimes, at least, enables one to find an optimal path and prove it so. In particular, it will be shown that a certain arrangement of 49 cities, one in each of the 48 states and Washington, D. C., is best, the $d_{ij}$ used representing road distances as taken from an atlas.
Now we will make advantage of these techniques to solve the TSP problem for 42 cities in the USA—using the Held-Karp algorithm would require $\sim 4$ hours (and unreasonable amounts of memory)!

1. Manchester, N. H.
4. Cleveland, Ohio
7. Indianapolis, Ind.
8. Chicago, Ill.
9. Milwaukee, Wis.
10. Minneapolis, Minn.
11. Pierre, S. D.
12. Bismarck, N. D.
13. Helena, Mont.
15. Portland, Ore.
16. Boise, Idaho
17. Salt Lake City, Utah
18. Carson City, Nev.
19. Los Angeles, Calif.
21. Santa Fe, N. M.
22. Denver, Colo.
23. Cheyenne, Wyo.
24. Omaha, Neb.
25. Des Moines, Iowa
26. Kansas City, Mo.
27. Topeka, Kans.
28. Oklahoma City, Okla.
29. Dallas, Tex.
30. Little Rock, Ark.
31. Memphis, Tenn.
32. Jackson, Miss.
33. New Orleans, La.
34. Birmingham, Ala.
35. Atlanta, Ga.
36. Jacksonville, Fla.
37. Columbia, S. C.
38. Raleigh, N. C.
40. Washington, D. C.
42. Portland, Me.
A. Baltimore, Md.
B. Wilmington, Del.
C. Philadelphia, Penn.
D. Newark, N. J.
E. New York, N. Y.
F. Hartford, Conn.
G. Providence, R. I.
### Demo: adjacency matrix

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>39 45</td>
<td></td>
</tr>
<tr>
<td>37 47</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>50 49</td>
<td>21 15</td>
<td></td>
</tr>
<tr>
<td>61 62</td>
<td>21 20 17</td>
<td></td>
</tr>
<tr>
<td>48 60</td>
<td>16 17 18 6</td>
<td></td>
</tr>
<tr>
<td>59 60</td>
<td>15 20 26 17 10</td>
<td></td>
</tr>
<tr>
<td>62 66</td>
<td>20 25 31 22 15 5</td>
<td></td>
</tr>
<tr>
<td>81 81</td>
<td>40 44 50 41 35 34 20</td>
<td></td>
</tr>
<tr>
<td>103 107</td>
<td>62 67 72 63 57 46 41 23</td>
<td></td>
</tr>
<tr>
<td>108 117</td>
<td>66 71 77 68 61 51 46 26 11</td>
<td></td>
</tr>
<tr>
<td>145 149</td>
<td>104 108 114 106 99 88 84 63 49 40</td>
<td></td>
</tr>
<tr>
<td>181 185</td>
<td>140 144 150 142 135 134 120 99 85 76 35</td>
<td></td>
</tr>
<tr>
<td>187 191</td>
<td>146 150 156 142 137 130 125 105 90 81 41 10</td>
<td></td>
</tr>
<tr>
<td>161 170</td>
<td>120 124 130 115 110 104 105 90 72 64 34 31 27</td>
<td></td>
</tr>
<tr>
<td>142 146</td>
<td>101 104 111 97 91 85 86 75 51 59 29 48 21</td>
<td></td>
</tr>
<tr>
<td>174 178</td>
<td>133 138 143 129 123 117 118 107 83 84 54 46 35 26 31</td>
<td></td>
</tr>
<tr>
<td>185 186</td>
<td>142 143 140 130 126 124 128 118 93 101 72 69 58 48 43 26</td>
<td></td>
</tr>
<tr>
<td>164 165</td>
<td>120 123 124 106 105 110 104 86 97 71 63 82 62 42 45 22</td>
<td></td>
</tr>
</tbody>
</table>

#### TABLE I

**Road Distances between Cities in Adjusted Units**

The figures in the table are mileages between the two specified numbered cities, less 11, divided by 17, and rounded to the nearest integer.
Demo: final solution

This tour has a length of 12,345 miles when the adjusted units are expressed in miles.

Fig. 16. The optimal tour of 49 cities.
The full implementation of this TSP solver in C++ (along with all the necessary files to perform this demo) may be found at:
https://github.com/PetarV-/Simplex-TSP-Solver

Methods similar to these have been successfully applied for solving far larger TSP instances. For example:
http://www.math.uwaterloo.ca/tsp/
13,509 largest towns in the US
15,112 largest towns in Germany
All 24,978 populated places in Sweden
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error \( \leq 0.05\% \)
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP

World TSP – 1,904,711 places, error ≤ 0.05%
World TSP – 1,904,711 places, error ≤ 0.05%
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error ≤ 0.05%
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error ≤ 0.05%
World TSP – 1,904,711 places, error $\leq 0.05\%$

Exactly solving TSP using the Simplex algorithm

Andrej Ivašković, Thomas Sauerwald
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error ≤ 0.05%
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error ≤ 0.05%
World TSP – 1,904,711 places, error ≤ 0.05%
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error ≤ 0.05%
World TSP – 1,904,711 places, error $\leq 0.05\%$
World TSP – 1,904,711 places, error ≤ 0.05%